

# Apple IIGS
# #16: Notes on Background Printing

Revised by: Mike Askins November 1988
Written by: Mike Askins June 1987

This Technical Note attempts to pinpoint some of the common problems people encounter when using background printing as available through the serial firmware.

## Calling Sequence

| | |
|---|---|
| `Init call` | Starts the serial firmware |
| `SetOutBuff` | Specifies a buffer to place data to be printed |
| | Places data in buffer ( amount < buffer size) |
| `SendQueue` | Starts the background printing process |

## Correctly Making the SendQueue Call

The *Apple IIGS Firmware Reference* incorrectly documents the parameters you pass to `SendQueue`. The correct specification of the recharge address does not correspond to the standard method of passing a full 32-bit address. Set the parameters as follows:

**SendQueue**
Launches background printing.

```
CmdList     DFB $04                    ;Parameter Count
            DFB$18                     ;Command Code
            DW $00                     ;Result Code (output)
            DW  DataLength
            DFB RechargeAddress (bank)
            DFB RechargeAddress (high)
            DFB RechargeAddress (low)
            DFB $00
```

## Using the Default Buffer

You can use the area which the firmware reserves for transparent buffering to place data for background printing. This is advantageous since the firmware calls the Memory Manager to

allocate space for the buffer (you must allocate the space from the Memory Manager if you use the `SetOutBuff` call to set up a buffer).

To use the serial firmware's buffer, you must first enable buffering by initializing the port with `PINIT` and sending it the string "^IBE" with `PWRITE`. Once you enable buffering, call `GetOutBuff` to find the size and location of the buffer, then place your data (buffersize - 1) in the buffer and call `SendQueue`.

## Data Size

Make sure that the amount of data you place in the buffer is at least one byte less than the size of the buffer since the firmware uses one byte of the buffer for bookkeeping purposes; if you place too much data in the buffer, it will continually print the buffer's contents and never call your recharge routine.

## The Recharge Routine

You should treat the recharge routine as an interrupt handler and execute it at interrupt time. Interrupts are disabled at this time, and it is illegal to enable them within the recharge routine. Like all interrupt handlers, the recharge routine should take care of its business as quickly as possible then exit; any excessive delays cause interrupt dependent processes (e.g., AppleTalk) to fail. You should also remember that most of the system code is non-reentrant; you should use the Scheduler when calling system code which may have been running when the serial interrupt that invoked the recharge routine occurred.

The serial firmware is not generally reentrant and does not interact with the Scheduler. If you want to make serial firmware calls (through $C1xx, $C2xx) from your recharge routine, you must preserve `MSLOT` (the byte at $0007F8) across those calls. Be aware that any non-recharge code must not make calls to the serial firmware that will disrupt the background printing process; sending the string "^BD" (disable buffering command), for example, is guaranteed to confuse a running background printing process.

### Further Reference

- *Apple IIGS Firmware Reference*